Electronic Version
Stylesheet Version v1.1.1

# Claims

[c1] 1. An intelligent platform management interface (IPMI) validating system optimally used between a host system having an IPMI, and an operating terminal, the IPMI validating system comprising:

a user interface generating output of a frame having a plurality of menus with optional items via the operating terminal wherein at least one of the optional items allows a user to load a test program for validating the IPMI;

an IPMI command engine module capable of directly encoding the loaded test program into IPMI commands and executing the IPMI commands;

an IPMI command management unit receiving the IPMI commands and transmitting each IPMI command to a channel assigned by the user; and

a channel management unit having a plurality of channel protocol conversion elements for transforming the IPMI command into a message conforming to the assigned channel and sending the message to the IPMI of the host system for validation.

[c2] 2. The IPMI validating system of claim 1 wherein one of the menus generated by the user interface includes at

least an open mode item for allowing the user to load a predefined test program, and an optional mode item for loading a default test program to implement fast validation.

[c3] 3. The IPMI validating system of claim 1 wherein the user interface further provides at least one channel item for the user to assign.

[c4] 4. The IPMI validating system of claim 2 wherein the default test program includes a system event log (SEL) test program, a watchdog test program, a sensor data record (SDR) test program, a chassis test program and a field replaceable unit (FRU) test program.

[c5] 5. The IPMI validating system of claim 1 wherein the IPMI command engine module is a compiled execution file written in a Delphi programming language.

[c6] 6. The IPMI validating system of claim 1 wherein the channel protocol conversion elements include at least a remote management control protocol (RMCP) element, an intelligent platform management bus (IPMB) protocol element, a keyboard control style interface (KCS) protocol element, and a universal asynchronous receiver/transmitter (UART) protocol element.

[c7] 7. An intelligent platform management interface (IPMI)

validating method optimally used between a host system having an IPMI and an operating terminal, the IPMI validating method comprising:

loading a test program from a menu generated by a user interface at the operating terminal to validate the IPMI;

encoding the loaded test program into an IPMI command by an IPMI command engine module and executing the IPMI command;

receiving the IPMI command and transmitting the IPMI command to a channel assigned by a user;

transforming the IPMI command into a message conforming to the protocol of the assigned channel by one of a plurality of channel protocol conversion elements; and

sending the message to the IPMI of the host system through the assigned channel for validation.

[c8] 8. The IPMI validating method of claim 7 further comprising loading at least one default test program by selecting a default item for validation.

[c9] 9. The IPMI validating method of claim 7 further comprising the user interface providing at least one available channel item for the user to assign.

[c10] 10. The IPMI validating method of claim 8 wherein the default test program includes a system event log (SEL)

test program, a watchdog test program, a sensor data record (SDR) test program, a chassis test program and a field replaceable unit (FRU) test program.

[c11] 11. The IPMI validating method of claim 7 wherein the channel protocol conversion elements include at least a remote management control protocol (RMCP) element, an intelligent platform management bus (IPMB) protocol element, a keyboard control style interface (KCS) protocol element, and a universal asynchronous receiver/transmitter (UART) protocol element.

[c12] 12. The IPMI validating method of claim 7 further comprising sending a corresponding validation result back from IPMI along said assigned channel to the user interface for output browsing and storing the result.